

Buyback Problem - Approximate matroid intersection with cancellation costs

Ashwinkumar B.V.

Cornell University, Ithaca, NY
ashwin85@cs.cornell.edu

Abstract. In the buyback problem, an algorithm observes a sequence of bids and must decide whether to accept each bid at the moment it arrives, subject to some constraints on the set of accepted bids. Decisions to reject bids are irrevocable, whereas decisions to accept bids may be canceled at a cost that is a fixed fraction of the bid value. Previous to our work, deterministic and randomized algorithms were known when the constraint is a matroid constraint. We extend this and give a deterministic algorithm for the case when the constraint is an intersection of k matroid constraints. We further prove a matching lower bound on the competitive ratio for this problem and extend our results to arbitrary downward closed set systems. This problem has applications to banner advertisement, semi-streaming, routing, load balancing and other problems where preemption or cancellation of previous allocations is allowed.

1 Introduction

Consider the online problem of resource allocation in which preemption is allowed. This kind of problem has been heavily studied in a wide variety of settings which range from advertisement allocations to routing to load balancing. In online weighted resource allocation without preemption we cannot get any non-trivial worst case guarantee on the sum of weights allocated. Consider the simplest problem of choosing the maximum number in a sequence. Any deterministic or randomized algorithm cannot have a constant competitive ratio up to any factor. Usually this impossibility is circumvented in the literature by placing some restrictions. This could be by allowing the input to be either a random permutation [1,2,3,4,5] or drawn iid from some probability distribution [6,7]. Other approach [8,9,10,11,12,13,14,15] which does not relax any conditions on the input assumes that either preemption is allowed or preemption with a penalty is allowed and gives guarantee for every input. In this paper we study this kind of relaxation.

Consider the following generic problem. There is a set system \mathcal{I} (downward closed) for the ground set \mathcal{E} . Elements from \mathcal{E} are presented to the algorithm in a sequential manner. Each element e_i is also associated with a utility w_{e_i} . When element e_i is presented to the algorithm it must be accepted or rejected immediately. When e_i is accepted the algorithm could cancel (preempt) some of the previously accepted elements. If S denotes the set currently accepted, then the constraint for the algorithm is to have $S \in \mathcal{I}$. The utility of the algorithm is the utility of the accepted elements minus the penalty paid to the canceled elements. All the canceled elements are paid a penalty proportional to their corresponding utility. We present some applications of this generic problem below.

Banner advertisement The buyback problem was first defined and studied in [8,9]. Specifically they give deterministic algorithms for the case when \mathcal{I} is a matroid. This was later extended by [16] which gave a randomized algorithm with better competitive ratio. Consider an advertisement system for a single advertisement slot. In certain systems bidding for this slot starts well in advance.

In such a system bidders come and bid in an online manner. The system accepts the bids or rejects them immediately. The system could later accept much higher bids and reject previously accepted ones. But this causes a loss for the previously accepted bidders. Hence the system pays them back with a penalty. The work in [8,9,16] also generalizes to much more general systems where the accepted bids can form a matroid. They leave open the question of finding algorithms for more general constraints. One of the key constraints not modeled by this is when each bidder desires a single item among a set of items, i.e when \mathcal{I} is a valid matching in a bipartite graph. Our result in section 2 solves this as well as a generalization to k arbitrary matroid constraints and our result in section 5 generalizes this to any downward closed set system. We also prove matching lower bounds in section 4. It is important to note that we ignore the incentives throughout our work and assume that bids/values are truthfully reported.

Free Disposal Consider the problem of online ad allocation with free disposal studied in [10]. Here we have a set of advertisers A known in advance together with an integer impression contract $n(a)$ for each advertiser $a \in A$. $n(a)$ denotes the maximum number of impressions for which advertiser can be charged. When an impression $i \in I$ arrives the utility of this impression w_{i_a} for every advertiser a is also revealed. The final utility of the algorithm is the total amount charged to each user. We note that there is a very straight forward reduction from this problem to the problem we define, specifically from the case when $k = 2$ and zero penalty for preemption ($f = 0$). Our algorithm gives a 5.828 competitive ratio. [10] gives an unconditional 2 competitive and conditional $e/(e - 1)$ competitive algorithm. Note that our algorithm is tight for the generic problem defined due to the lower bound shown in section 4. [10] is able to give better competitive ratio as this problem is more restrictive than the generic problem we define.

Semi Streaming Recently few papers [17,18,19,20,21,22,23] have studied graph problems in the semi-streaming model. Consider the problem of finding a weighted matching in a stream which uses $\tilde{O}(n)$ memory, $O(1)$ update time and 1 pass. [23] introduced this problem and gave a factor 6 approximation. [17] improved this to 5.828 approximate semi-streaming algorithm. Both of these algorithms are characterized by the fact that they always maintain a valid matching. Hence our lower bound in section 4 proves that among the class of algorithms which maintain only edges of a valid matching no algorithm can achieve better than approximation ratio 5.828.

Routing with preemption and Load Balancing There has been a huge literature [11,12,13,14,15] on Routing in Networks and Load Balancing where preemption of previously allocated resources is allowed. Many of these results studied can be very succinctly generalized by the following problem.

Consider an arbitrary downward closed set system \mathcal{I} . Elements $e_i \in \mathcal{I}$ are presented to the algorithm along with its weight and the sets to which it belongs to. The algorithm has to either accept or reject the element immediately. An accepted element can be preempted or canceled later but canceled/rejected element cannot be taken later.

This problem is precisely the problem we study with the restriction of penalty for cancellation being 0. As we will show in section 5, such a generic problem does not have any algorithm with competitive ratio better than $n - 1$ even when penalty is 0. Also a trivial extension of the algorithm from [8] will give a competitive ratio of $(n-1)(1+2f+2\sqrt{f(1+f)})$. One should note that the papers which study routing with preemption and load balancing are able to achieve better competitive ratio by exploiting some additional structure in the problem.

The offline problem of matroid intersection was introduced and studied in [24,25]. The problem has also been studied under more general submodular utility functions in the CS theory/OR literature. Some of the recent papers in this direction are [26,27,28,29].

Our Contributions The algorithm we propose is a greedy type algorithm which is a natural generalization of the algorithm from [8,9]. While the algorithm is simple to state the analysis turns out to be much harder. There are two key technical contributions in this paper.

1. The main technical hurdle in analyzing the algorithm comes from bounding the utility of the final accepted set S as compared to the optimal set OPT (Lemma 1). For this we develop a new novel type of charging scheme. This charging scheme is also aided by a graph construction which could be of independent interest in discrete optimization.
2. We also give optimal lower bounds. The main technical hurdle in this case is identifying the input on which the algorithm has worst case behavior. There is a key difference in the lower bound for $k = 1$ case (given in [8]) and $k = 2$ case. The general k is quite similar to $k = 2$. The matroid used in the lower bound for $k = 1$ case in [8] is just a uniform matroid of rank 1. For the case $k = 2$ we need a intersection of two partition matroids (i.e matching) which has a lot more structure. This is used to derive an infinite sequence $\{x_1, x_2, \dots\}$ with certain properties.

We also note here the portions of the paper which follow more easily from [8]. In the case of analyzing the algorithm the part which bounds the penalty uses a geometric series argument and is quite akin to [8]. Similarly in the lower bound once the sequence $\{x_1, x_2, \dots\}$ along with its properties are identified the rest of the analysis follows more easily.

2 Preliminaries

First we define the problem formally

2.1 Model

Consider a ground set of Elements $\mathcal{E} = \{e_1, e_2, \dots, e_n\}$. Let $\mathcal{M}_1, \dots, \mathcal{M}_k$ ¹ be k arbitrary matroid constraints on \mathcal{E} . Let the corresponding Independent sets be $\mathcal{I}_1, \dots, \mathcal{I}_k$ and let $\mathcal{I} = \cap_{j=1}^k \mathcal{I}_j$. We define the online problem with the following constraints.

1. The elements of \mathcal{E} are presented to the algorithm in some arbitrary order. The value w_{e_i} of element e_i and the matroid constraints it is involved with are revealed to the algorithm when the element is presented to it.
2. When element e_i is presented it must be accepted or rejected immediately. Additionally it could be canceled at a later point in time. When an element is canceled the algorithm must pay a penalty $f \cdot w_{e_i}$ where f is a constant. (Note the difference between reject and cancel)
3. Let \mathcal{A} be the set of elements accepted and \mathcal{R} be the set of elements accepted and later canceled. Then the utility of the algorithm is defined as $\sum_{e \in \mathcal{A}} w_e - (1 + f) \sum_{e \in \mathcal{R}} w_e$. Note that all elements in \mathcal{R} are also counted in \mathcal{A} . Moreover the currently maintained set $S = \mathcal{A} - \mathcal{R}$ must be an independent set i.e $S \in \mathcal{I}$.

Here we desire to find a competitive online algorithm with the above constraints.

¹ A matroid $\mathcal{M}_i = (\mathcal{E}, \mathcal{I}_i)$ is constructed from a ground set $\mathcal{E} \neq \phi$ and a nonempty family of subsets of \mathcal{E} , called the independent subsets of \mathcal{E} , such that if $B \in \mathcal{I}_i$ and $A \subseteq B$ then $A \in \mathcal{I}_i$ (\mathcal{I}_i is hereditary). Additionally, if $A, B \in \mathcal{I}_i$ and $|A| < |B|$, then there is some element $x \in B - A$ such that $A \cup \{x\} \in \mathcal{I}_i$ (exchange property).

Algorithm 1: Online Matroid Intersection:

```

1: Initialize  $S = \emptyset$ .
2: for all elements  $e_i$ , in order of arrival, do
3:   if  $S \cup \{e_i\} \in \mathcal{I}$  then
4:      $S = S \cup \{e_i\}$ 
5:   else
6:     for all  $1 \leq j \leq k$  do
7:       if  $S \cup \{e_i\} \notin \mathcal{I}_j$  then
8:          $e_{i_j}$  be the element of smallest value such
           that  $S \cup \{e_i\} \setminus \{e_{i_j}\} \in \mathcal{I}_j$ 
9:       else
10:         $e_{i_j} = NULL$ .
11:       end if
12:     end for
13:     Let  $C_{e_i} = \cup_{j=1}^k \{e_{i_j}\}$ 
14:     if  $w_{e_i} \geq r \cdot (\sum_{j=1}^k w_{e_{i_j}})$  then
15:        $S = S \cup \{e_i\} \setminus \cup_{j=1}^k \{e_{i_j}\}$ 
16:     end if
17:   end if
18: end for

```

Algorithm 2: Online Matroid Intersection:

```

1: Initialize  $S = \emptyset$ .
2: for all elements  $e_i$ , in order of arrival, do
3:   if  $S \cup \{e_i\} \in \mathcal{I}$  then
4:      $S = S \cup \{e_i\}$ 
5:   else
6:      $S' = \text{Greedy on } S \cup \{e_i\}$  a
7:     Let  $C_{e_i} = S \cup \{e_i\} - S'$  (Rejected Elements)
8:     if  $w_{e_i} \geq r \cdot (\sum_{e \in C_{e_i}} w_e)$  then
9:        $S = S'$ 
10:    end if
11:  end if
12: end for

```

^a Here Greedy means sorting and choosing the max weight elements satisfying the matroid constraints

Fig. 1. Algorithms for k matroids intersection**2.2 Algorithm**

The algorithm is shown in Figure 1 as Algorithm 1. At each step the algorithm maintains an Independent set S . Assume it sees the element e_i at some step. If $S \cup \{e_i\}$ is also an independent set, then it includes $\{e_i\}$ into the current set S . Otherwise $S \cup \{e_i\}$ has a circuit in some of the matroids \mathcal{I}_j . It first finds the minimum value element (e_{i_j}) it must remove in set S to make $S \cup \{e_i\}$ an independent set in each of \mathcal{I}_j . Now suppose $w_{e_i} \geq r \cdot (\sum_{j=1}^k w_{e_{i_j}})$, then it includes the element e_i and discards the elements $\cup_{j=1}^k \{e_{i_j}\}$. We will prove that the above algorithm is $\frac{(k \cdot r - 1)r}{r - 1 - f}$ competitive. Here r is a constant defined later to optimize the competitive ratio.

The Algorithm 2 in Figure 1 is an equivalent formulation of Algorithm 1. It is not difficult to show that steps 6-12 of Algorithm 1 are equivalent to step 6 of Algorithm 2. For ease of analysis we will just analyze Algorithm 1.

3 Analysis of the algorithm

Let $S(i)$ be the set S at the end of step i and let $OPT \subseteq \mathcal{E}$ be optimal solution to the weighted intersection of k matroids. The main part of competitive analysis is based on the following lemma.

Lemma 1. $w(S(n))^{\frac{(k \cdot r - 1)r}{r - 1}} \geq w(OPT)$ where $w(S(n)) = \sum_{e \in S(n)} w_e$.

We will prove Lemma 1 in 3.1. The proof is based on a new type of charging scheme. For now we just assume it to analyze the competitive ratio of the algorithm. We once again note that the main technical contribution is in analyzing Lemma 1 and given Lemma 1 the analysis of Theorem 1 (i.e bounding the penalty) is very similar to [8].

Theorem 1. *The online algorithm with cancellations for k matroid constraints has a competitive ratio $c = \frac{(k \cdot r - 1)r}{r - 1 - f}$. This ratio is minimized when $\frac{r}{1+f} = 1 + \sqrt{1 - \frac{1}{k(1+f)}}$ and has a value $c = k(1+f)(1 + \sqrt{1 - \frac{1}{k(1+f)}})^2$*

Proof. The competitive ratio of our algorithm matches the case $k = 1$ given in [8,9]. Later in section 4 we will show that this is tight for every k .

The utility of the algorithm comprises of two terms. One is due to utility of $S(n)$ and the other is the penalty due to canceled set R .

- For each element e_i we define a value $P(e_i)$ recursively. If e_i was accepted in step 3 or was never accepted, then $P(e_i) = 0$. Else if elements $C_{e_i} = \{e_{i_1}, e_{i_2}, \dots, e_{i_k}\}$ were canceled, then $P(e_i) = f \cdot \sum_{j=1}^k w_{e_{i_j}} + \sum_{j=1}^k P(e_{i_j})$. Now each canceled item e_{i_j} 's penalty is accounted to the item e_i which canceled it. We prove that for any element e_i the total penalty accounted is less than or equal to $\frac{f}{r-1} \cdot w_{e_i}$. The proof is by induction. The base case when $P(e_i) = 0$ is simple. The inductive case is as follows.

$$\begin{aligned}
P(e_i) &= f \cdot \sum_{j=1}^k w_{e_{i_j}} + \sum_{j=1}^k P(e_{i_j}) \\
&\leq f \cdot \sum_{j=1}^k w_{e_{i_j}} + \sum_{j=1}^k f \cdot \frac{w_{e_{i_j}}}{r-1} \\
&= \frac{fr}{r-1} \cdot \sum_{j=1}^k w_{e_{i_j}} \\
&\leq \frac{fr}{r-1} \cdot \frac{w_{e_i}}{r} \\
&= f \cdot \frac{w_{e_i}}{r-1}
\end{aligned} \tag{1}$$

Hence the total penalty is at-most $\sum_{e_i \in S(n)} f \cdot \frac{w_{e_i}}{r-1} = \frac{f \cdot w(S(n))}{r-1}$

- The final weight of the set $S(n)$ is bounded by Lemma 1. Combining the two parts we get the total utility of the algorithm.

$$\begin{aligned}
\text{Utility} &\geq w(S(n)) - f \cdot \frac{w(S(n))}{r-1} \\
&= \frac{r-1-f}{r-1} \cdot w(S(n)) \\
&\geq \frac{r-1-f}{(k \cdot r - 1)r} \cdot w(OPT) \quad (\text{Using Lemma 1})
\end{aligned} \tag{2}$$

Hence we get competitive ratio of $c = \frac{(k \cdot r - 1)r}{r - 1 - f}$. Optimizing over r we get $\frac{r}{1+f} = 1 + \sqrt{1 - \frac{1}{k(1+f)}}$ and $c = k(1+f)(1 + \sqrt{1 - \frac{1}{k(1+f)}})^2$. \square

3.1 Charging scheme

Here we will prove Lemma 1. This portion is technically the hardest part of the paper and requires developing a new charging scheme. This is aided by a graph construction. We first give some notation. Each element carries two kinds of charges (ch_1 and ch_2). Let at any step j , $\text{ch}(e_i, j) = \text{ch}_1(e_i, j) + \text{ch}_2(e_i, j)$. Additionally let $C_{e_i} = \cup_{j=1}^k e_{i_j}$ (the set of elements discarded when e_i is included) be as defined in step 13 of Algorithm 1. Let $S(j)$ denote the set S after step j . Let $\text{ch}(S', j) = \sum_{e \in S'} \text{ch}(e, j)$ (analogously for ch_1 and ch_2).

Sketch We start with a total charge of OPT on the elements. $ch_1(e_i, j)$ denotes the charge which the element carries from the beginning. $ch_2(e_i, j)$ denotes the charge which the elements gets from some other element. At any step either e_i or C_{e_i} is discarded. When any element is discarded all its charge is re-added to $S(i)$ (or $S(j)$ for $j \geq i$). There by all the charge is stored in $S(n)$. Next we bound the amount of charge any element can carry.

There are two ways charge is transferred. One way is for ch_1 and another for ch_2 . At step i if C_{e_i} is discarded, then we always transfer $ch_2(C_{e_i}, i-1)$ to $ch_2(e_i, i)$. Another way of transfer is for ch_1 . This is done by a fairly sophisticated graph construction. Essentially we construct k bipartite graphs and then prove that each one has a matching that matches all vertices on the left side of the bipartition using Hall's Theorem. Suppose e_i is matched to e_j , we transfer $ch_1(e_i, a)$ to $ch_2(e_j, b)$. Here a denotes the step at which e_i was removed and b denotes the step before which e_j was removed (or otherwise). Note that we will need causal consistency ($a < b$) for this.

The charges at the beginning of the algorithm are defined as follows.

- if $e_i \in OPT$ then $ch(e_i, 0) = ch_1(e_i, 0) = w_{e_i}$ and $ch_2(e_i, 0) = 0$
- if $e_i \notin OPT$ then $ch(e_i, 0) = ch_1(e_i, 0) = ch_2(e_i, 0) = 0$.

Before defining the charging scheme we define a graph construction which will aid us in the charging scheme.

Graph Construction Construct k bi-partite graphs as the algorithm proceeds. Here p^{th} graph corresponds to the p^{th} matroid. Let $P_1(p)$ denote partite set 1 of p^{th} graph and $P_2(p)$ denote partite set 2 of p^{th} graph. Additionally let $N_p(S)$ denote the set of neighbors of $S \subseteq P_1(p)$ in p^{th} graph. Let $rank_p(S)$ be the rank of set S in the p^{th} matroid.

1. The bi-partite graph starts empty and edges are added. Each end point of an edge corresponds to an element e_i . The node corresponding to an element e_i exists only when corresponding edge is added and removed when all its adjacent edges are deleted. An edge in the graph corresponds to a potential ch_1 transfer.
2. Consider step 14 in the algorithm. If $w_{e_i} < r \cdot \sum_{j=1}^k w_{e_{i_j}}$, then e_i is not included in S . Now if $e_i \in OPT$, then add a node e_i to $P_1(p)$ (for each p). Let $Ckt(e_i, p)$ be the unique circuit in p^{th} matroid in $S \cup \{e_i\}$. Then add edge e_i, e_j for each $e_j \in Ckt(e_i, p) - \{e_i\}$ with e_j belonging to $P_2(p)$.
3. Consider step 14 in the algorithm. If $w_{e_i} \geq r \cdot \sum_{j=1}^k w_{e_{i_j}}$, then $C_i = \{e_{i_1}, \dots, e_{i_k}\}$ is deleted from S and e_i is included into it. Delete each e_{i_p} from the corresponding $P_2(p)$ ². For each existing edge e_q, e_{i_p} add edges e_q, e_j for each $e_j \in Ckt(e_i, p) - \{e_{i_p}\}$ with e_j belonging to $P_2(p)$. Additionally if $e_{i_p} \in OPT$ (i.e $ch_1(e_{i_p}, 0) > 0$), then re-add it to $P_1(p)$. Add edges e_{i_p}, e_j for each $e_j \in Ckt(e_i, p) - \{e_{i_p}\}$.

Lemma 2. *The graph construction has the following properties.*

1. $P_1(p) \subseteq OPT - S(n)$ for each p .
2. $\forall \hat{S} \subseteq P_1(p), N_p(\hat{S})$ spans $\hat{S} \subseteq P_1(p)$ in p^{th} matroid.
3. $\forall \hat{S} \subseteq P_1(p), |N_p(\hat{S}) - OPT| \geq |\hat{S}|$.
4. There exists a matching in graph p such that every $e \in P_1(p)$ is matched to a node in $P_2(p) - OPT$.

² Note that e_{i_p} is deleted only from $P_2(p)$ and not from $P_2(p')$ for $p' \neq p$

5. Any element $e \in \mathcal{E} - S(n) - OPT$ is matched in at-most $k - 1$ of the graphs from the side of P_2 .

Proof. 1. This is easily seen by construction. In steps 2 and 3 of Graph construction an element is added to $P_1(p)$ precisely when it is removed from S and when it belongs to OPT .

2. When an node e_j is added to $P_1(p)$ then edges to each element in $Ckt - \{e_j\}$ are added. Hence any node e_j is spanned by $N_p(\{e_j\})$. By matroid property this implies that for any set $\hat{S} \subseteq P_1(p)$ we have that $N_p(\hat{S})$ spans \hat{S} .³

3. Let $W = N_p(\hat{S}) \cap OPT$. We now assert some statements from which the inequality easily follows.

- $rank_p((N_p(\hat{S}) - OPT) \cup W) \geq rank_p(\hat{S} \cup W)$ ⁴. This follows from property 2.
 - $rank_p(\hat{S} \cup W) = rank_p(\hat{S}) + rank_p(W) = |\hat{S}| + |W|$. This follows from the fact that \hat{S} and W are disjoint and $\hat{S} \cup W \subseteq OPT$.
 - $rank_p((N_p(\hat{S}) - OPT) \cup W) \leq rank_p(N_p(\hat{S}) - OPT) + rank_p(W) \leq |N_p(\hat{S}) - OPT| + |W|$.
- These set of inequalities follows from the matroid property.

Combining the above equations we get $|\hat{S}| \leq |N_p(\hat{S}) - OPT|$.

4. Follows from Hall's Theorem and property 3. If $e_i \in P_1(p)$ is matched to e_j , then let $e_j = M_p(e_i)$.

5. This follows from step 3 of graph construction. Here any element removed from $S(i)$ in any step is deleted from $P_2(p)$ (for one of the p 's). Hence such an element could belong to P_2 of at-most $k - 1$ graphs and be matched at-most $k - 1$ times (from P_2 's side). Note that any element $e \in OPT - S(n)$ could additionally be matched from the P_1 's side. □

Charge Transfer We finally explain the exact way the charge is transferred in each step.

- Consider step 14 in the algorithm. If $w_{e_i} \geq r \cdot (\sum_{j=1}^k w_{e_{i_j}})$, then transfer all of $ch_2(e_{i_j}, i - 1)$ for each e_{i_j} to $ch_2(e_i, i)$.
- Consider step 14 in the algorithm. If $w_{e_i} \geq r \cdot (\sum_{j=1}^k w_{e_{i_j}})$, then let $C_{e_i} = \{e_{i_1}, \dots, e_{i_k}\}$ be deleted from S and e_i is included into it. If e_{i_l} was re-added to $P_1(l)$, then transfer all of $ch_1(e_{i_l}, 0)$ to $ch_2(M_l(e_{i_l}), t)$ (where t is either the step $M_l(e_{i_l})$ is deleted or n).
- Consider step 14 in the algorithm. Let $w_{e_i} < r \cdot (\sum_{j=1}^k w_{e_{i_j}})$. Additionally if $e_i \in OPT$, then it would have been added to $P_1(l)$ of each graph l . Now e_i is matched to different nodes in different graphs. Transfer a portion of $ch_1(e_i, 0)$ to $ch_2(M_l(e_i), t)$ which is proportional to $w_{e_{i_l}}$ for each graph l . (where t is either the step $M_l(e_i)$ is deleted or n).
- Note that the above transfer of charges does not violate causal consistency as the transfer of charge happens from e to some element in $S(i)$ of the future.

We finish the proof of Lemma 1 by analyzing the charge transfer. First note that any element in $\mathcal{E} - S(n)$ receives ch_1 transfer in step 2 or 3 of charge transfer at-most $k - 1$ times. This is by property 5 of Lemma 2. Additionally we can also see that each ch_1 transfer to element e_i is at-most $r \cdot w_{e_i}$. Using these properties by Induction that we prove that $ch_2(e_i, j) \leq \frac{(k-1)r^2}{r-1} w_{e_i}$ for $e_i \in \mathcal{E} - S(n)$ and $ch(e_i, j) \leq \frac{(k \cdot r - 1)r}{r-1} w_{e_i}$ for $e_i \in S(n)$.

³ Note that even though the edges could later be deleted, the span property still holds due to additional edges being added.

⁴ $rank_p(S)$ is defined as the largest subset $A \subseteq S$ such that $A \in \mathcal{I}_p$

- For any element in $e_i \in \mathcal{E} - S(n)$ we have $\text{ch}_2(e_i, j) \leq \frac{(k-1)r^2}{r-1} w_{e_i}$ if it was deleted at step j . Note that ch_1 transfer happens atmost $k-1$ times for $e_i \in \mathcal{E} - S(n) - OPT$ and 0 times for $e_i \in \mathcal{E} - S(n) \cap OPT$

$$\begin{aligned}
\text{ch}_2(e_i, j) &\leq (k-1) \cdot (\text{ch}_1 \text{ transfer}) + \text{ch}_2 \text{ transfer} \\
&\leq (k-1) \cdot r \cdot w_{e_i} + \sum_{j=1}^k \text{ch}_2(e_{i_j}, i-1) \\
&\leq (k-1) \cdot r \cdot w_{e_i} + \sum_{j=1}^k \frac{(k-1)r^2}{r-1} w_{e_{i_j}} \\
&= (k-1) \cdot r \cdot w_{e_i} + \frac{(k-1)r^2}{r-1} \sum_{j=1}^k w_{e_{i_j}} \\
&\leq (k-1) \cdot r \cdot w_{e_i} + \frac{(k-1)r^2}{r-1} \frac{w_{e_i}}{r} \\
&= \frac{(k-1)r^2}{r-1} w_{e_i}
\end{aligned} \tag{3}$$

- For any element in $S(n) - OPT$ we have $\text{ch}(e_i, n) \leq \frac{(k \cdot r - 1)r}{r-1} w_{e_i}$. Note that this is also true for any element in $S(n) \cap OPT$, as they do not get any ch_1 transfer but have a non-zero ch_1 .

$$\begin{aligned}
\text{ch}(e_i, n) &\leq k \cdot (\text{ch}_1 \text{ transfer}) + \text{ch}_2 \text{ transfer} \\
&\leq k \cdot r \cdot w_{e_i} + \sum_{j=1}^k \text{ch}_2(e_{i_j}, i-1) \\
&\leq k \cdot r \cdot w_{e_i} + \sum_{j=1}^k \frac{(k-1)r^2}{r-1} w_{e_{i_j}} \\
&= k \cdot r \cdot w_{e_i} + \frac{(k-1)r^2}{r-1} \sum_{j=1}^k w_{e_{i_j}} \\
&\leq k \cdot r \cdot w_{e_i} + \frac{(k-1)r^2}{r-1} \frac{w_{e_i}}{r} \\
&= \frac{(k \cdot r - 1)r}{r-1} w_{e_i}
\end{aligned} \tag{4}$$

The above argument proves that

- $\forall e_i \in S(n), \text{ch}(e_i, n) \leq \frac{(k \cdot r - 1)r}{r-1} w(e_i)$
- $\sum_{e_i \in S(n)} \text{ch}(e_i, n) = w(OPT)$. This follows naturally from charge conservation in the system.

The proof of Lemma 1 can be easily seen from the above two properties.

4 Lower Bound

Sketch Here we prove a matching lower bound of $c = k(1+f)(1+\sqrt{1-\frac{1}{k(1+f)}})^2$ for the competitive ratio. Assume \mathcal{A} is an online deterministic algorithm which achieves a competitive ratio $\beta < c$. Then we arrive at a contradiction. The proof will be in following steps.

1. We will construct a k -dimensional matching. Using this we will argue the existence of an infinite sequence $X = \{x_1, x_2, \dots\}$ of the following form. $x_1 = 1$ and $x_i > 0, \forall i$. Additionally they will satisfy the following inequality.

$$\beta(x_i - f \cdot \sum_{j=1}^{i-1} x_j) \geq x_{i+1} + (k-1) \sum_{j=1}^{i+1} x_j, \forall i \geq 1 \tag{5}$$

2. Consider any sequence $X = \{x_1, x_2, \dots\}$ which satisfies $x_i > 0, \forall i$ and $\beta(x_i - f \cdot \sum_{j=1}^{i-1} x_j) \geq x_{i+1} + (k-1) \sum_{j=1}^{i+1} x_j, \forall i$. Now if $\beta < c$, we arrive at a contradiction.

We once again note that the important new idea here is in first step. The first step needs to get the exact matroid structures right so as to pin the tight inequality. The lower bound given in [8] for $k = 1$ is just a uniform matroid of rank 1 while for $k > 1$ we need intersection of partition matroids which has a lot more structure. Given the part 1 in our construction part 2 follows more easily from techniques in [8]. We will prove the second part first.

4.1 Contradiction

Consider all sequences of the form $x_1 = 1, x_i > 0, \forall i$ and satisfying equation 5. We first claim that if a sequence of the above form exists, then there should exist a sequence where inequality 5 is equality $\forall i$. Assume by contradiction that such a sequence does not exist. For any given sequence X , let $n(X)$ be the minimum i for which inequality 5 is strict. Among the sequences consider sequence X for which $n(X) = N$ is as large as possible. We construct a sequence for which $n(X)$ is even larger thus arriving at a contradiction. Let λ be defined as follows.

$$\lambda = \frac{\beta(x_N - f \cdot \sum_{j=1}^{N-1} x_j) - (k-1) \sum_{j=1}^N x_j}{k \cdot x_{N+1}} \quad (6)$$

Let $X' = \{x_1, x_2, \dots, x_N, \lambda x_{N+1}, \lambda_{N+2} x_{N+2}, \dots\}$. Then it is easy to see that inequality 5 and other constraints are met. Additionally $n(X') > n(X)$. Hence we arrive at a contradiction to the non-existence of a sequence when inequality 5 is equality $\forall i$. Let $Y = \{y_1, y_2, \dots\} (y_i \geq 0)$ be the sequence such that

$$y_1 = 1, \quad \beta(y_i - f \cdot \sum_{j=1}^{i-1} y_j) = y_{i+1} + (k-1) \sum_{j=1}^{i+1} y_j, \forall i \geq 1 \quad (7)$$

Let $z_i = \sum_{j=1}^i y_j$. Then we get the recurrence $k \cdot z_{i+1} = (1 + \beta)z_i - \beta(1 + f)z_{i-1}$ and $z_1 = 1$. As each $y_i \geq 0$ we also have that $z_i \geq 0$. Now we use a Lemma from [30] to get a contradiction.

Lemma 3. [30] *Let $u_n = au_{n-1} + bu_{n-2}$ be a linear recurrence of second order. If $p(x) = x^2 - ax - b$ has imaginary roots, then the sequence must have negative elements.*

Consider the case when $\beta < k(1 + f)(1 + \sqrt{1 - \frac{1}{k(1+f)}})^2$. Then it is simple to see that $D = (1 + \beta)^2 - 4 \cdot k \cdot \beta(1 + f) < 0$ which implies that $k \cdot x^2 = (1 + \beta)x - \beta(1 + f)$ has imaginary roots. Hence by Lemma 3 this implies that the sequence $\{z_i\}$ has negative elements which is a contradiction to our assumption.

4.2 Construction of sequence

Given the online algorithm with competitive ratio β we construct k dimensional matching using which we will construct the sequence $\{x_i\}$. For ease of description we will restrict to the case $k = 2$. The general k case is similar. In other words we construct bi-partite graphs. Here each partite corresponds to a partition matroid. Hence the subset of edges is in the intersection of the two matroids if and only if it forms a valid matching. The graph will have two types of edges ex_i and ey_i . We will use x_i and y_i to denote the corresponding weights of the edges.

- Start with edge ex_1 with weight $x_1 = 1$.

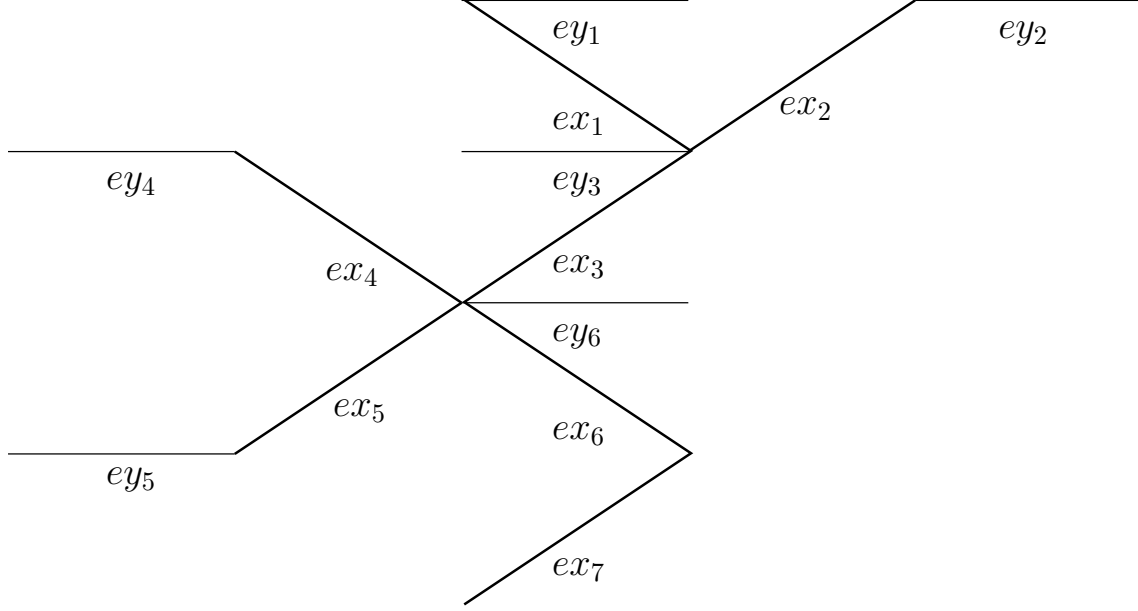


Fig. 2. Example k dimensional matching. Step before adding ey_7, ex_8

- For simplicity we just state the inductive step in the construction. At step i the edge held by the algorithm is ex_i (and no other edge).
- At step $i + 1$ we add edges to both ends of ex_i such that they differ in weight by at most ϵ and the algorithm accepts exactly 1 of them. Due to the matching condition it has to reject the currently accepted edge ex_i . Here adding a new edge to one end of ex_i means a edge is revealed. This new edge shares one vertex with ex_i and the other vertex is a brand new vertex which hasn't yet appeared in the algorithm. We describe how this is done below.
At step $i + 1$ add edges to both ends of ex_i of weight ϵ . If the algorithm does not accept either of the new edges rewind the algorithm and instead add edges of weight 2ϵ . Do this rewind, add edges of higher weights (higher by ϵ) till the algorithm accepts exactly 1 new edge. Due to the matching constraint this means the current edge ex_i sharing an end point must be canceled and a penalty paid to it. Let the accepted edge be named ex_{i+1} and the other newly added edge be named ey_i . By construction we have $x_{i+1} \leq y_i + \epsilon$.
- Use the above construction to construct an infinite sequence x_i and y_i . An example construction is shown in figure 2.

We will note some properties of the sequence x_i and y_i .

- At step $i + 1$ the algorithm accepts ex_{i+1} and cancels ex_i while not accepting ey_i . Consider the rewinding procedure in which the algorithm is presented edge e' of weight $x_{i+1} - \epsilon$ and ey_i before ex_{i+1} is presented. By the construction of our rewinding procedure both e' and ey_i would not be accepted and ex_i would still be the currently maintained edge in the solution. In such a case we assert some statements based on which we derive an inequality.
 - The utility of the algorithm is $x_i - f \sum_{j=1}^{i-1} x_{i-1}$ as ex_i is the currently maintained edge and $\{ex_1, \dots, ex_{i-1}\}$ are currently canceled edges.

- It is clear that $\{ey_1, ey_2, \dots, ey_i, e'\}$ is a valid matching in the current set of revealed edges. This has total weight $x_{i+1} - \epsilon + \sum_{j=1}^i y_j$
- By definition of β we have $\beta(x_i - f \sum_{j=1}^{i-1} x_{i-1}) \geq x_{i+1} - \epsilon + \sum_{j=1}^i y_j$.
- By construction of the sequence we also know that $y_i + \epsilon \geq x_{i+1}$.

Substituting we get $\beta(x_i - f \sum_{j=1}^{i-1} x_{i-1}) \geq x_{i+1} - \epsilon + \sum_{j=2}^{i+1} x_j - (i+1)\epsilon$. Tending ϵ to 0⁵ we get $\beta(x_i - f \sum_{j=1}^{i-1} x_{i-1}) \geq x_{i+1} + \sum_{j=2}^{i+1} x_j$

- Note that the sequence constructed is not the one we desired. The sum on the right hand side starts from $j = 2$. $\beta(x_i - f \sum_{j=1}^{i-1} x_{i-1}) \geq x_{i+1} + \sum_{j=2}^{i+1} x_j$ implies $\beta(x_i - f \sum_{j=2}^{i-1} x_{i-1}) \geq x_{i+1} + \sum_{j=2}^{i+1} x_j$. So from the sequence $\{x_1, x_2, \dots\}$ deleting x_1 and rescaling x_2 to 1 we get the desired sequence.

The general case k is very similar to case $k = 2$ but instead involves construction of k -dimensional matching.

5 Extending to arbitrary downward closed set systems.

We extend the algorithm in section 2 to arbitrary downward closed set systems. One way is to represent the set system as a intersection of k matroids and use the algorithm in previous sections. But even the algorithm for single item from [8] gives a competitive ratio $n \cdot (1 + 2f + 2\sqrt{f(1+f)})$. A simple modification can be used to improve it to $(n-1)(1 + 2f + 2\sqrt{f(1+f)})$. As we show next this is essentially the best that can be done.

Theorem 2. *Even for the case $f = 0$ the competitive ratio cannot be better than $n - 1$.*

Proof. The proof is by constructing a set system for which every algorithm achieves competitive ratio worse than $n - 1$. Consider the downward closed set system \mathcal{I} which is the set of independent sets in a graph. Now we construct the graph as follows. At every step a node in the graph arrives and reveals its weight and edges to existing set of vertices. First node N_1 and N_2 with weights 1 arrive and have a edge between them. Without loss of generality assume that the algorithm \mathcal{A} accepts N_1 and rejects N_2 . Next at every step a node N_i arrives with weight $1 - \epsilon$ and edge to N_1 . The algorithm can never reject N_1 and accept a new N_i without violating the fact that its competitive ratio is less than $n - 1$. But at the end $OPT = \{N_2, N_3, \dots, N_n\}$ with weight $1 + (n-2)(1 - \epsilon)$ while the weight held by algorithm is 1. Hence the competitive ratio is at-least $1 + (n-2)(1 - \epsilon)$. Tending $\epsilon \rightarrow 0$ we get that competitive ratio is at-least $n - 1$. □

6 Acknowledgments

The author thanks Robert D. Kleinberg, Renato Paes Leme and Hu Fu for useful comments and suggestions on an earlier draft of the paper.

⁵ We are ignoring some issues of convergence for ease of exposition

References

1. Babaioff, M., Immorlica, N., Kleinberg, R.: Matroids, secretary problems, and online mechanisms. In: SODA. (2007) 434–443
2. Babaioff, M., Immorlica, N., Kempe, D., Kleinberg, R.: A knapsack secretary problem with applications. In: APPROX-RANDOM. (2007) 16–28
3. Dynkin, E.B.: Optimal choice of the stopping moment of a Markov process. Dokl. Akad. Nauk SSSR **150** (1963) 238–240
4. Korula, N., Pál, M.: Algorithms for secretary problems on graphs and hypergraphs. In: ICALP '09: Proceedings of the 36th International Colloquium on Automata, Languages and Programming, Berlin, Heidelberg, Springer-Verlag (2009) 508–520
5. Kleinberg, R.: A multiple-choice secretary algorithm with applications to online auctions. In: SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms, Philadelphia, PA, USA, Society for Industrial and Applied Mathematics (2005) 630–631
6. Hajiaghayi, M.T., Kleinberg, R., Sandholm, T.: Automated online mechanism design and prophet inequalities. In: AAAI'07: Proceedings of the 22nd national conference on Artificial intelligence, AAAI Press (2007) 58–65
7. Chawla, S., Hartline, J.D., Malec, D.L., Sivan, B.: Multi-parameter mechanism design and sequential posted pricing. In: STOC '10: Proceedings of the 42nd ACM symposium on Theory of computing, New York, NY, USA, ACM (2010) 311–320
8. Babaioff, M., Hartline, J.D., Kleinberg, R.: Selling ad campaigns: online algorithms with buyback. In: Proc. 10th ACM Conference on Electronic Commerce. (2009)
9. Constantin, F., Feldman, J., Muthukrishnan, S., Pál, M.: An online mechanism for ad slot reservations with cancellations. In: Proc. 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). (2009) 1265–1274
10. Feldman, J., Korula, N., Mirrokni, V., Muthukrishnan, S., Pál, M.: Online ad assignment with free disposal. In: WINE '09: Proceedings of the 5th International Workshop on Internet and Network Economics, Berlin, Heidelberg, Springer-Verlag (2009) 374–385
11. Adler, R., Azar, Y.: Beating the logarithmic lower bound: randomized preemptive disjoint paths and call control algorithms. In: SODA '99: Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms, Philadelphia, PA, USA, Society for Industrial and Applied Mathematics (1999) 1–10
12. Canetti, R., Irani, S.: Bounding the power of preemption in randomized scheduling. In: STOC '95: Proceedings of the twenty-seventh annual ACM symposium on Theory of computing, New York, NY, USA, ACM (1995) 606–615
13. Azar, Y., Blum, A., Mansour, Y.: Combining online algorithms for rejection and acceptance. In: SPAA '03: Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures, New York, NY, USA, ACM (2003) 159–163
14. Garay, J.A., Gopal, I.S.: Call preemption in communication networks. In: IEEE INFOCOM '92: Proceedings of the eleventh annual joint conference of the IEEE computer and communications societies on One world through communications (Vol. 3), Los Alamitos, CA, USA, IEEE Computer Society Press (1992) 1043–1050
15. Garay, J.A., Gopal, I.S., Kutten, S., Mansour, Y., Yung, M.: Efficient on-line call control algorithms. J. Algorithms **23** (1997) 180–194
16. Ashwinkumar, B.V., Kleinberg, R.: Randomized online algorithms for the buyback problem. In: WINE '09: Proceedings of the 5th International Workshop on Internet and Network Economics, Berlin, Heidelberg, Springer-Verlag (2009) 529–536
17. McGregor, A.: Finding graph matchings in data streams. In: APPROX-RANDOM. (2005) 170–181
18. Feigenbaum, J., Kannan, S., McGregor, A., Suri, S., Zhang, J.: Graph distances in the streaming model: the value of space. In: SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms, Philadelphia, PA, USA, Society for Industrial and Applied Mathematics (2005) 745–754
19. Demetrescu, C., Finocchi, I., Ribichini, A.: Trading off space for passes in graph streaming problems. In: SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm, New York, NY, USA, ACM (2006) 714–723
20. Cormode, G., Muthukrishnan, S.: Space efficient mining of multigraph streams. In: PODS '05: Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, New York, NY, USA, ACM (2005) 271–282
21. Das Sarma, A., Gollapudi, S., Panigrahy, R.: Estimating pagerank on graph streams. In: PODS '08: Proceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, New York, NY, USA, ACM (2008) 69–78

22. Buchsbaum, A.L., Giancarlo, R., Racz, B.: New results for finding common neighborhoods in massive graphs in the data stream model. *Theor. Comput. Sci.* **407** (2008) 302–309
23. Feigenbaum, J., Kannan, S., McGregor, A., Suri, S., Zhang, J.: On graph problems in a semi-streaming model. In Diaz, J., Karhumki, J., Lepist, A., Sannella, D., eds.: *Automata, Languages and Programming*. Volume 3142 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg (2004) 17–44
24. Jenkyns, T.A.: The efficacy of the greedy algorithm. In: *Proc. of 7th South Eastern Conference on Combinatorics, Graph Theory and Computing*. (1976) 341–350
25. Korte, B., Hausmann, D.: An analysis of the greedy heuristic for independence systems. In: *Annals of Discrete Math.* (1978) 2:65–74
26. Feige, U., Mirrokni, V.S., Vondrak, J.: Maximizing non-monotone submodular functions. In: *FOCS '07: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, Washington, DC, USA, IEEE Computer Society (2007) 461–471
27. L., N.G., A., W.L., L., F.M.: An analysis of approximations for maximizing submodular set functions ii. In: *Mathematical Programming Study*. (1978) 73–87
28. Reichel, J., Skutella, M.: Evolutionary algorithms and matroid optimization problems. In: *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, New York, NY, USA, ACM (2007) 947–954
29. Lee, J., Mirrokni, V.S., Nagarajan, V., Sviridenko, M.: Non-monotone submodular maximization under matroid and knapsack constraints. In: *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, New York, NY, USA, ACM (2009) 323–332
30. Halava, V., Harju, T., Hirvensalo, M.: Positivity of second order linear recurrent sequences. *Discrete Appl. Math.* **154** (2006) 447–451